

BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO CUỐI KỲ CÔNG NGHỆ MỚI TRONG PHÁT TRIỂN ỨNG DỤNG CNTT

ĐỀ TÀI:

**Hệ thống giám sát và cảnh báo thông qua thiết bị di động
dựa trên cảm biến**

Giảng viên hướng dẫn : ThS. Nguyễn Ngọc Lễ
Nhóm thực hiện : Nhóm 11
Lớp học phần : DHKHMT16A - 420300314701

Thành phố Hồ Chí Minh, tháng 11 năm 2023

ĐẠI HỌC CÔNG NGHIỆP TP. HCM

KHOA CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh Phúc

Tp. Hồ Chí Minh, tháng 11 năm 2023.

BÁO CÁO CUỐI KỲ

MÔN CÔNG NGHỆ MỚI TRONG PHÁT TRIỂN ỨNG DỤNG CNTT

ĐỀ TÀI:

Hệ thống giám sát và cảnh báo thông qua thiết bị di động dựa trên cảm biến

Giảng viên hướng dẫn : ThS. Nguyễn Nọc Lễ

Nhóm thực hiện : Nhóm 11

Lớp học phần: : DHKHMT16A – 420300314701

Thành viên trong nhóm:

20014891	Nguyễn Thành An
20019721	Bùi Thị Yến Yến
20065451	Diêm Công Bình

Mục Lục

Mục Lục	3
I. Giới thiệu	5
II. Cơ sở lý thuyết	5
1. Adafruit IO	5
1.1. Adafruit IO là gì?	5
1.2. Adafruit IO có thể làm gì?	5
2. Message Queuing Telemetry Transport(MQTT)	6
2.1. MQTT là gì?	6
2.2. Các thành phần của MQTT	6
2.3. Tổng quan cách thức hoạt động của MQTT	7
3. Teachable Machine	8
4. Linh kiện điện tử	8
4.1. Cảm biến khí hóa lỏng, khí than MQ-5.....	8
4.2. Module Cảm Biến Độ Ẩm, Nhiệt Độ DHT11	10
4.3. Arduino UNO R3 DIP.....	11
4.4. Led phủ màu	12
5. Android Studio	13
III. Hiện thực	14
1. Adafruit IO	14
1.1. Feeds	14
1.2. Dashboard	14
2. Mô hình nhận diện	15
2.1. Dữ liệu.....	15
2.2. Hiện thực bằng Python.....	17
3. Đấu nối.....	18
4. Android Studio	21
4.1. Yêu cầu thiết bị	21
4.2. Hiện thực bằng code	22
IV. Đánh giá và tổng kết	24
1. Mô hình nhận dạng	24
2. Kết quả hiện thực	24
V. Ứng dụng và hướng phát triển	25

Tài liệu tham khảo:	26
---------------------------	----

I. Giới thiệu

Trong thời đại công nghệ hiện đại, việc phát triển ứng dụng di động đã trở thành một môn công nghệ mới đầy hứa hẹn. Đề tài này tập trung vào việc xây dựng một hệ thống đầy có thể sẽ hữu ích: một hệ thống giám sát và cảnh báo thông qua thiết bị di động, nhằm mang đến sự an toàn và tiện ích cho gia đình.

Với ý tưởng này, chúng tôi sẽ kết nối giữa các thiết bị cảm biến trong nhà và các thiết bị di động thông qua công nghệ Adafruit IO. Điều này cho phép thu thập dữ liệu quan trọng về khí ga, nhiệt độ và độ ẩm từ môi trường xung quanh, và truyền các thông tin này lên thiết bị di động.

Tuy nhiên, hệ thống không chỉ đơn thuần là một công cụ giám sát môi trường. Nó còn có khả năng phát hiện sự xuất hiện của con người trong phạm vi giám sát. Bằng cách sử dụng một mô hình nhận dạng được huấn luyện trên Teachable Machine sử dụng TensorFlow.js, một thư viện học máy trong Javascript, để huấn luyện. Sẽ sử dụng mã Python để tích hợp mô hình nhận dạng vào ứng dụng, tạo ra khả năng nhận diện người thời gian thực. Hệ thống có thể nhận diện và thông báo cho chúng ta về sự có mặt của người khác trong nhà.

Với ứng dụng được phát triển bằng Android Studio, sẽ tạo ra một giao diện di động thân thiện và trực quan, giúp người dùng dễ dàng theo dõi và kiểm soát môi trường sống của mình.

Tóm lại, đề tài này mang đến một giải pháp hiện đại và thông minh cho việc giám sát và cảnh báo trong gia đình. Từ dữ liệu thu thập từ các cảm biến và thông qua Adafruit IO, để có thể đảm bảo an toàn và tiện ích cho gia đình, với khả năng nhận diện nguy hiểm và sự hiện diện của con người trong thời gian thực.

II. Cơ sở lý thuyết

1. Adafruit IO

1.1. Adafruit IO là gì?

Adafruit IO là một dịch vụ đám mây do Adafruit Industries cung cấp. Adafruit IO cung cấp một cách thuận tiện để kết nối và giao tiếp với các thiết bị Internet of Things (IoT) thông qua Internet.

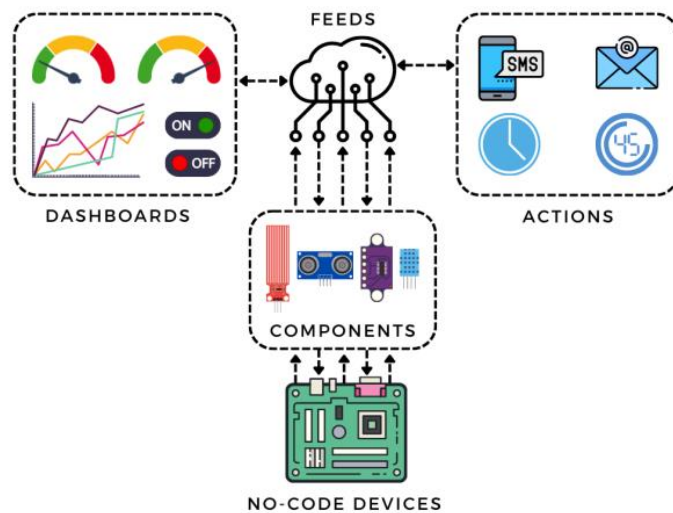
1.2. Adafruit IO có thể làm gì?

Adafruit IO cho phép người dùng tạo và quản lý các "thiết bị ảo" (virtual devices) và các "kênh" (feeds) để thu thập và lưu trữ dữ liệu từ các thiết bị IoT.

Đồng thời, điều khiển động cơ, đọc dữ liệu từ cảm biến, hiển thị dữ liệu và tương tác với các thiết bị IoT trực tuyến và theo thời gian thực.

Giúp hỗ trợ nhiều giao thức và giao diện lập trình để truyền và nhận dữ liệu, bao gồm MQTT, RESTful API và giao diện web.

Với nó, người dùng có thể dễ dàng xây dựng ứng dụng IoT, kết nối các dự án với các dịch vụ web như Twitter, RSS feeds, dịch vụ thời tiết, v.v. kết nối dự án với các thiết bị hỗ trợ kết nối internet khác để phân tích và sử dụng.



Ảnh 1. Adafruit IO can do

Tất cả những gì đã được nêu trên đều có thể thực hiện miễn phí với Adafruit IO.

2. Message Queuing Telemetry Transport(MQTT)

2.1. MQTT là gì?

MQTT là một giao thức nhắn tin dựa trên các tiêu chuẩn hoặc một bộ các quy tắc được sử dụng cho việc giao tiếp máy với máy.

Cảm biến thông minh, thiết bị đeo trên người và các thiết bị Internet vạn vật (IoT) khác thường phải truyền và nhận dữ liệu qua mạng có tài nguyên và băng thông hạn chế. Các thiết bị IoT này sử dụng MQTT để truyền dữ liệu vì giao thức này dễ triển khai và có thể giao tiếp dữ liệu IoT một cách hiệu quả. MQTT hỗ trợ nhắn tin giữa các thiết bị với đám mây và từ đám mây đến thiết bị.

2.2. Các thành phần của MQTT

MQTT triển khai mô hình xuất bản/đăng ký bằng cách định nghĩa máy chủ và trình truyền tải như:

Máy khách MQTT:

Một máy khách MQTT là bất kỳ thiết bị nào từ một máy chủ đến một bộ vi điều khiển có chạy một thư viện MQTT. Nếu máy khách gửi thông điệp, nó hoạt động như một nơi gửi thông điệp và nếu nhận thông điệp, nó hoạt động như bên nhận. Về cơ bản, bất kỳ thiết bị nào giao tiếp bằng MQTT qua một mạng đều có thể được gọi là một thiết bị khách MQTT.

Trình truyền tải MQTT:

Trình truyền tải MQTT là hệ thống backend điều phối thông điệp giữa các máy khách khác nhau. Trách nhiệm của trình truyền tải bao gồm nhận và lọc thông điệp, xác định máy khách đã đăng ký nhận từng thông điệp và gửi thông điệp cho các máy khách đó. Trình trung chuyển cũng chịu trách nhiệm thực hiện các tác vụ khác như:

- Ủy quyền và xác thực máy khách MQTT
- Chuyển thông điệp đến các hệ thống khác để phân tích thêm
- Xử lý các thông điệp bị bỏ lỡ và các phiên trên máy khách

Kết nối MQTT:

Máy khách và máy chủ bắt đầu giao tiếp bằng một kết nối MQTT. Máy khách khởi tạo kết nối bằng cách gửi một thông điệp CONNECT (KẾT NỐI) đến trình truyền tải MQTT. Trình truyền tải xác nhận rằng kết nối đã được thiết lập bằng cách trả lời bằng một thông điệp CONNACK. Cả máy khách MQTT và trình truyền tải đều cần đến ngăn xếp TCP/IP để giao tiếp. Các máy khách không bao giờ kết nối với nhau mà chỉ kết nối với trình truyền tải.

2.3. Tổng quan cách thức hoạt động của MQTT

Giao thức MQTT hoạt động trên nguyên tắc của mô hình Publish/ Subscribe(Xuất bản/ Đăng ký) . Trong quá trình giao tiếp mạng truyền thống, máy khách và máy chủ giao tiếp trực tiếp với nhau. Máy khách yêu cầu tài nguyên hoặc dữ liệu từ máy chủ, sau đó máy chủ xử lý và gửi lại một phản hồi.

Tuy nhiên, MQTT sử dụng mẫu xuất bản/đăng ký để phân tách bên gửi thông điệp (nơi gửi thông điệp) khỏi bên nhận thông điệp (nơi nhận thông điệp). Thay vào đó, một thành phần thứ ba được gọi là trình truyền tải thông điệp xử lý quá trình giao tiếp giữa nơi gửi thông điệp và nơi nhận thông điệp. Việc của trình truyền tải là lọc tất cả các thông điệp đến từ nơi gửi thông điệp và phân phối các thông điệp đó một cách chính xác đến nơi nhận thông điệp.

Tóm lại, đầu tiên máy khách MQTT thiết lập kết nối với trình truyền tải MQTT. Sau khi được kết nối, máy khách có thể xuất bản thông điệp, đăng ký nhận các thông điệp cụ thể hoặc thực hiện cả hai. Cuối cùng, khi trình truyền tải MQTT nhận được một thông điệp, trình truyền tải sẽ chuyển thông điệp đó đến những nơi nhận thông điệp quan tâm.

Chủ đề MQTT:

Thuật ngữ "chủ đề" đề cập đến từ khóa mà MQTT sử dụng để lọc thông điệp cho máy khách. Chúng được tổ chức theo thứ bậc, tương tự như đường dẫn đến tệp hoặc thư mục. *Ví dụ: Một hệ thống nhà thông minh có thể có các chủ đề như "nhà/phòng khách/đèn" hoặc "nhà/phòng bếp/nhiệt độ."*

Quá trình Xuất bản và Đăng ký qua MQTT:

Xuất bản (Publish) qua MQTT: Máy khách MQTT xuất các thông điệp có chứa chủ đề và dữ liệu cụ thể dưới định dạng byte. Máy khách xác định định dạng dữ liệu như dữ liệu văn bản, dữ liệu nhị phân, tệp XML hoặc JSON. *Ví dụ: đèn phòng khách đã được bật.*

Đăng ký (Subscribe) qua MQTT: Máy khách MQTT gửi một thông điệp SUBSCRIBE (ĐĂNG KÝ) đến trình truyền tải MQTT để nhận thông điệp về các chủ đề quan tâm. Thông điệp này chứa một mã định danh duy nhất và một danh sách đăng ký. *Ví dụ: ứng dụng nhà thông minh trên điện thoại đăng ký chủ đề "đèn/phòng khách" để theo dõi trạng thái đèn.*

3. Teachable Machine

Teachable Machine là một công cụ trực tuyến giúp tạo ra các mô hình học máy một cách nhanh chóng, dễ dàng và dễ tiếp cận cho mọi người.

Teachable Machine Community sử dụng mô hình máy học được huấn luyện bằng thư viện Tensorflow.js. Tensorflow.js là một thư viện máy học trong ngôn ngữ JavaScript, cho phép huấn luyện và triển khai mô hình máy học trực tiếp trong trình duyệt web.

Trong quá trình sử dụng Teachable Machine, người dùng có thể huấn luyện máy tính nhận diện hình ảnh, âm thanh và tư thế mà không cần phải viết mã máy học. Nó giúp làm giảm độ phức tạp của việc huấn luyện mô hình bằng cách cung cấp một giao diện trực quan và thân thiện với người dùng, thích hợp cho người dùng không có kiến thức trước về học máy.

Khi người dùng tạo và huấn luyện mô hình trong Teachable Machine, mô hình đó sau đó có thể được xuất ra và tích hợp vào các dự án, trang web, ứng dụng, và nhiều hơn nữa thông qua các thư viện và đoạn mã hỗ trợ được cung cấp trong repository của Teachable Machine Community.

4. Lĩnh kiện điện tử

4.1. Cảm biến khí hóa lỏng, khí than MQ-5



Figure 2. MQ-5 LPG & Gas Detector Sensor Module

Đặc tính thiết bị

- Nguồn cung cấp: 2.5V ~ 5V.
- Tích hợp MQ-5 gas Sensor.
- Kích thước : 40mm * 21mm.
- Led báo hiệu.

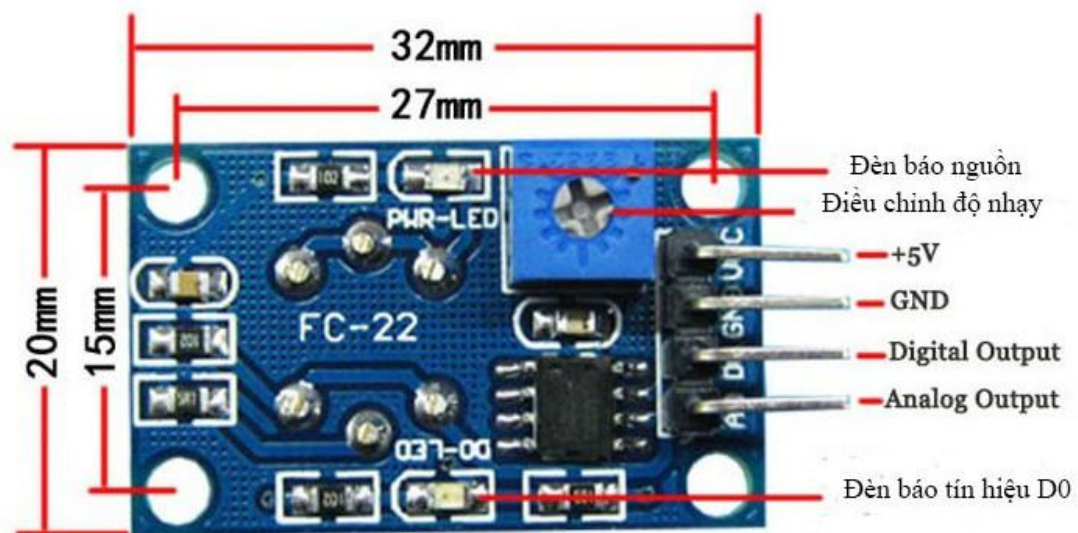


Figure 3. Sơ đồ mạch MQ-5

Nguyên lý hoạt động

Khi cảm biến hoạt động nó sẽ truyền tín hiệu từ các chân DOUT và AOUT của mình về vi điều khiển.

- Tín hiệu DOUT:
 - o Tín hiệu thấp : có khí gas.
 - o Tín hiệu cao : không có khí gas.
- Tín hiệu AOUT: cho tín hiệu tương tự.

Và khi có khí gas 2 đèn LED trên module sẽ phát sáng.

Ứng dụng minh họa: Thiết bị cảnh báo khi có khí Gas.

Chuẩn bị phần cứng

- 1 Arduino Uno R3.
- 1 Module khí Gas (MQ – 5 Gas Sensor).
- Phụ kiện: loa, đèn led, trở, dây kết nối...

4.2. Module Cảm Biến Độ Ẩm, Nhiệt Độ DHT11

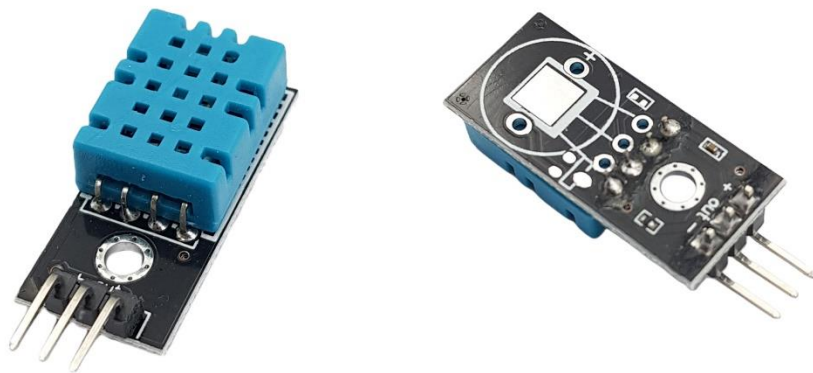


Figure 4. DHT11 Temperature and Humidity Sensor Modul

Thông số kỹ thuật

- Điện áp hoạt động: 5VDC
- Chuẩn giao tiếp: TTL, 1 wire.
- Khoảng đo độ ẩm: 20%-80%RH sai số $\pm 5\%RH$
- Khoảng đo nhiệt độ: 0-50°C sai số $\pm 2^{\circ}C$
- Tần số lấy mẫu tối đa 1Hz (1 giây / lần)
- Kích thước: 28mm x 12mm x10m

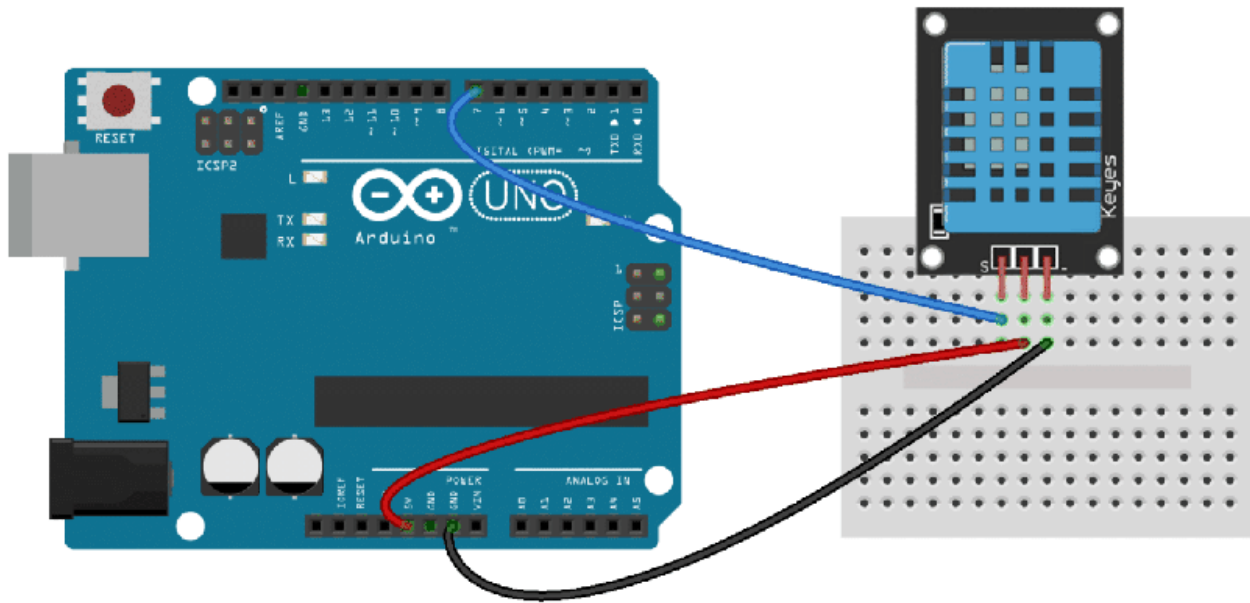


Figure 5. Sơ đồ kết nối cảm biến độ ẩm và nhiệt độ DHT11

4.3. Arduino UNO R3 DIP

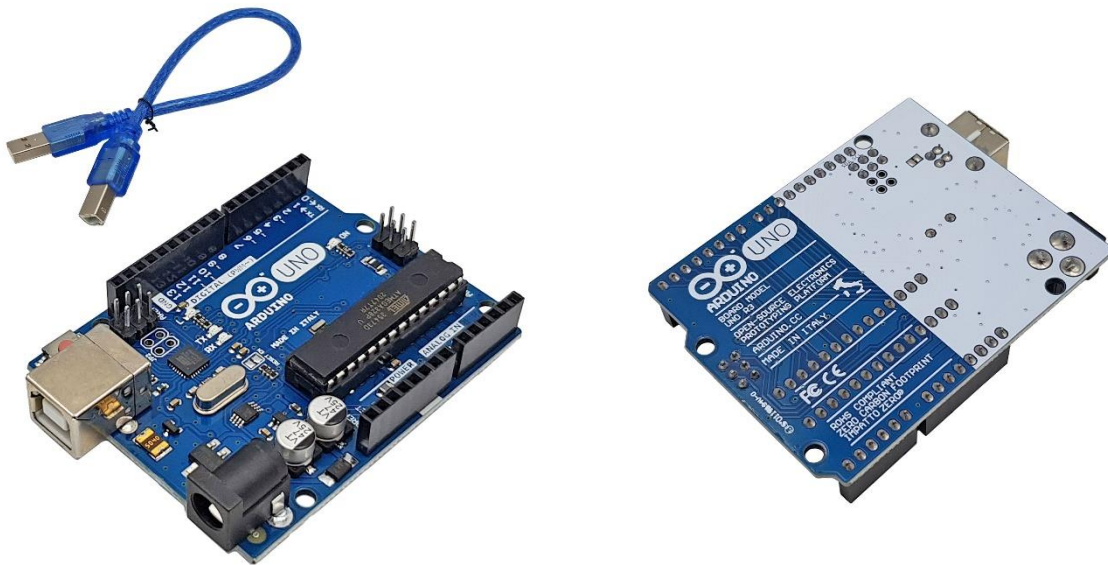


Figure 6. Arduino UNO R3 DIP_1

Thông số kỹ thuật:

- Vi điều khiển ATmega328 họ 8bit
- Điện áp hoạt động 5~12V DC (khuyến dùng)
- Tần số hoạt động 16 MHz

- Dòng tiêu thụ Khoảng 30mA
- Điện áp vào giới hạn 19V DC
- Số chân Digital I/O 14 (6 chân PWM)
- Số chân Analog 6 (độ phân giải 10bit)
- Dòng tối đa trên mỗi chân I/O 30 mA
- Dòng ra tối đa (5V) 500 mA
- Dòng ra tối đa (3.3V) 50 mA
- Bộ nhớ flash 32 KB (ATmega328) với 0.5KB dùng bởi bootloader
- SRAM 2 KB (ATmega328)
- EEPROM 1 KB (ATmega328)
- Khối lượng 25 gram

Lưu ý: Cấp điện áp trên 5.5V vào các chân Digital hoặc Analog của Arduino UNO sẽ làm hỏng vi điều khiển.

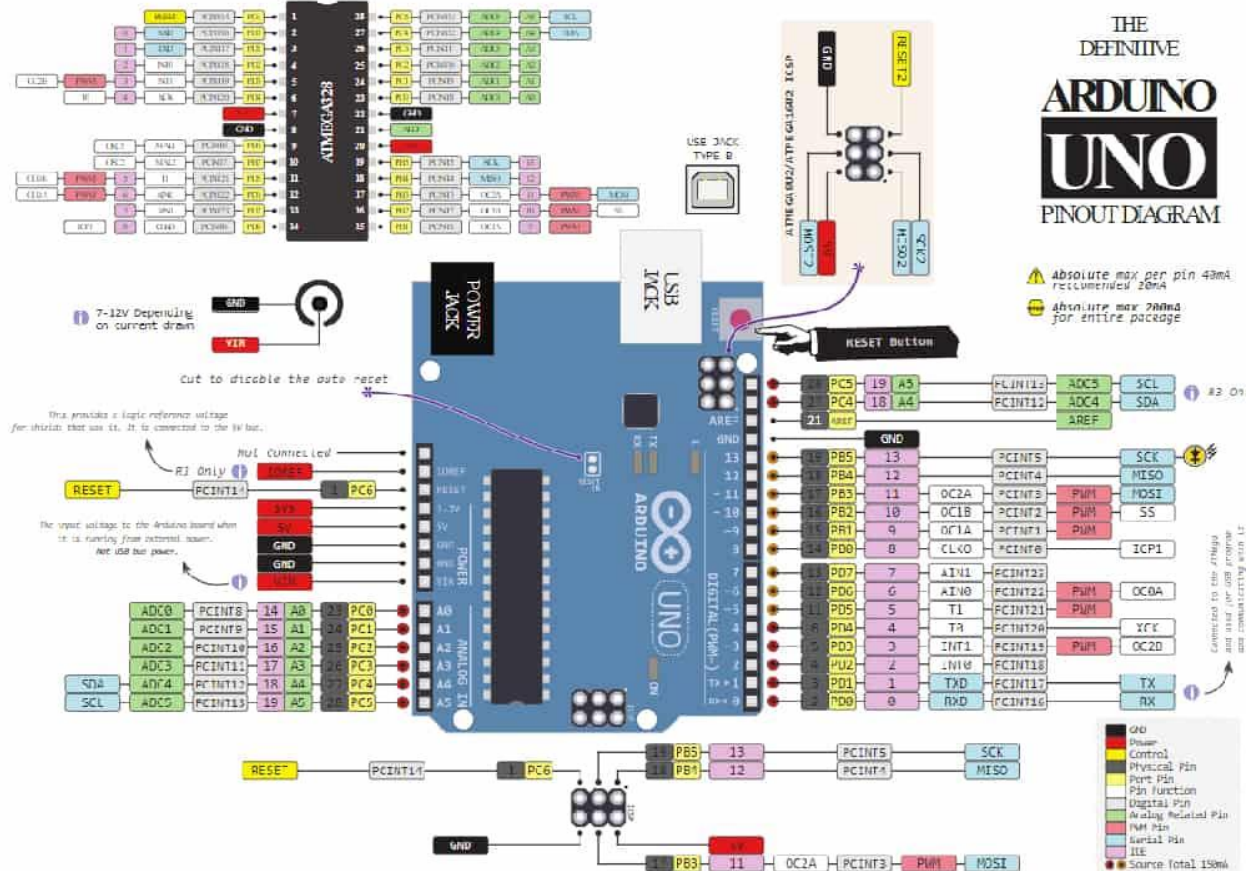


Figure 7. Sơ đồ chân

4.4. Led phủ màu



Figure 8. Led phủ màu 3mm

Thông số kỹ thuật:

- Loại led: led phủ
- Kích thước: 3mm
- Điện áp: 1.9 – 2.1 V
- Dòng: 10 – 20 mA
- Độ sáng: 3000 – 4000 MCD
- Trọng lượng: ~ 2g (10 led)

5. Android Studio

Android Studio là Môi trường phát triển tích hợp (IDE) chính thức để phát triển ứng dụng Android. Nhờ có công cụ cho nhà phát triển và trình soạn thảo mã mạnh mẽ của IntelliJ IDEA, Android Studio cung cấp thêm nhiều tính năng giúp bạn nâng cao năng suất khi xây dựng ứng dụng Android, chẳng hạn như:

- Một hệ thống xây dựng linh hoạt dựa trên Gradle
- Một trình mô phỏng nhanh và nhiều tính năng
- Một môi trường hợp nhất nơi bạn có thể phát triển cho mọi thiết bị Android
- Tính năng Live Edit (Chỉnh sửa trực tiếp) để cập nhật các thành phần kết hợp trong trình mô phỏng và thiết bị thực theo thời gian thực
- Mã mẫu và quá trình tích hợp GitHub để giúp bạn xây dựng các tính năng ứng dụng phổ biến cũng như nhập mã mẫu
- Đa dạng khung và công cụ thử nghiệm
- Công cụ tìm lỗi mã nguồn (lint) để nắm bắt hiệu suất, khả năng hữu dụng, khả năng tương thích với phiên bản và các vấn đề khác
- Hỗ trợ C++ và NDK
- Tích hợp sẵn tính năng hỗ trợ Google Cloud Platform, giúp dễ dàng tích hợp Google Cloud Messaging và App Engine

III. Hiện thực

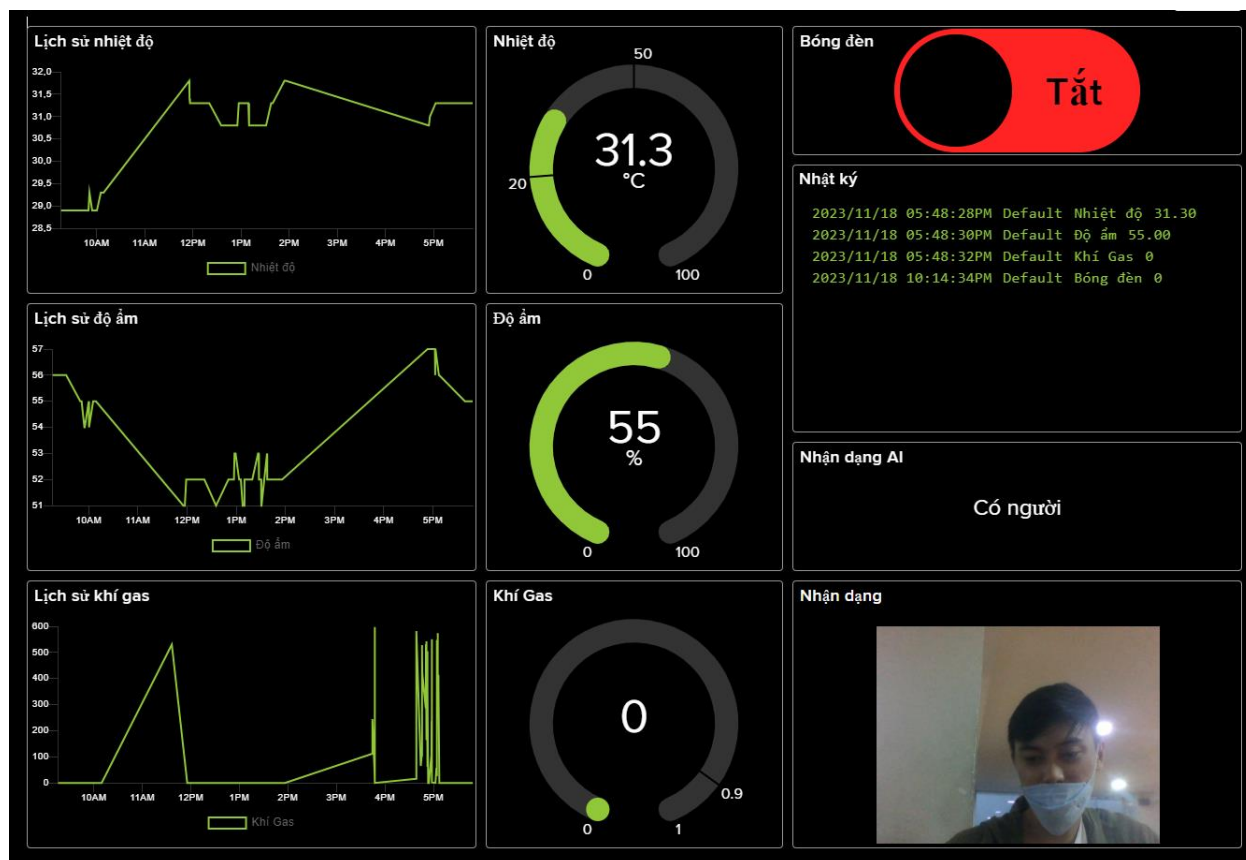
1. Adafruit IO

1.1. Feeds

Default			
Feed Name	Key	Last value	Recorded
<input type="checkbox"/> Bóng đèn	bong-den	0	about 1 hour ago
<input type="checkbox"/> Hình ảnh	hinh-anh	/9j/4AAQSkZJRgABAQAA...	about 6 hours ago
<input type="checkbox"/> Khí Gas	khi-gas	0	about 5 hours ago
<input type="checkbox"/> Nhiệt độ	nhiet-do	31.30	about 5 hours ago
<input type="checkbox"/> Nhận dạng	nhan-dang	Có người	about 6 hours ago
<input type="checkbox"/> ai	ai	Không có người	1 day ago
<input type="checkbox"/> khi-gá	khi-ga	444	about 8 hours ago
<input type="checkbox"/> light	light	1	2 days ago
<input type="checkbox"/> Độ ẩm	do-am	55.00	about 5 hours ago
<input type="checkbox"/> Ảnh	anh	/9j/4AAQSkZJRgABAQAA...	about 7 hours ago

1.2. Dashboard

Dashboards		
<input type="checkbox"/> Name	Key	Created At
<input type="checkbox"/> IOT	iot	October 11, 2023



2. Mô hình nhận diện

2.1. Dữ liệu

Thu thập 362 mẫu hình ảnh không có người và 1082 mẫu hình ảnh có người từ Kaggle và ảnh thực tế.

Tiến hành đào tạo mô hình nhận diện bằng Teachable Machine với:

- Epochs: 100 (đào tạo qua toàn bộ tập dữ liệu 100 lần)
- Batch Size: 16 (xem xét và cập nhật trọng số dựa trên 16 mẫu dữ liệu mỗi lần)
- Learning Rate: 0.001 (độ lớn của các bước cập nhật trọng số)

Thu được mô hình với kết quả như sau:








Teachable Machine

Không có người ✎

362 Image Samples

Webcam

Upload










Có người ✎

1082 Image Samples

Webcam

Upload



Add a class

Training

Model Trained

Advanced

Epochs: 100

Batch Size: 16

Learning Rate: 0.001


Reset Defaults

Under the hood

Preview

Export Model

Input ON Webcam



Output

Không có người

Có người 99%








Teachable Machine

Không có người ✎

362 Image Samples

Webcam

Upload










Có người ✎

1082 Image Samples

Webcam

Upload



Add a class

Training

Model Trained

Advanced

Epochs: 100

Batch Size: 16

Learning Rate: 0.001

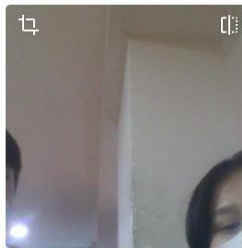
Reset Defaults

Under the hood

Preview

Export Model

Input ON Webcam



Output

Không có người

Có người 100%








Teachable Machine

Không có người ✎

362 Image Samples

Webcam

Upload










Có người ✎

1082 Image Samples

Webcam

Upload



Add a class

Training

Model Trained

Advanced

Epochs: 100

Batch Size: 16

Learning Rate: 0.001


Reset Defaults

Under the hood

Preview

Export Model

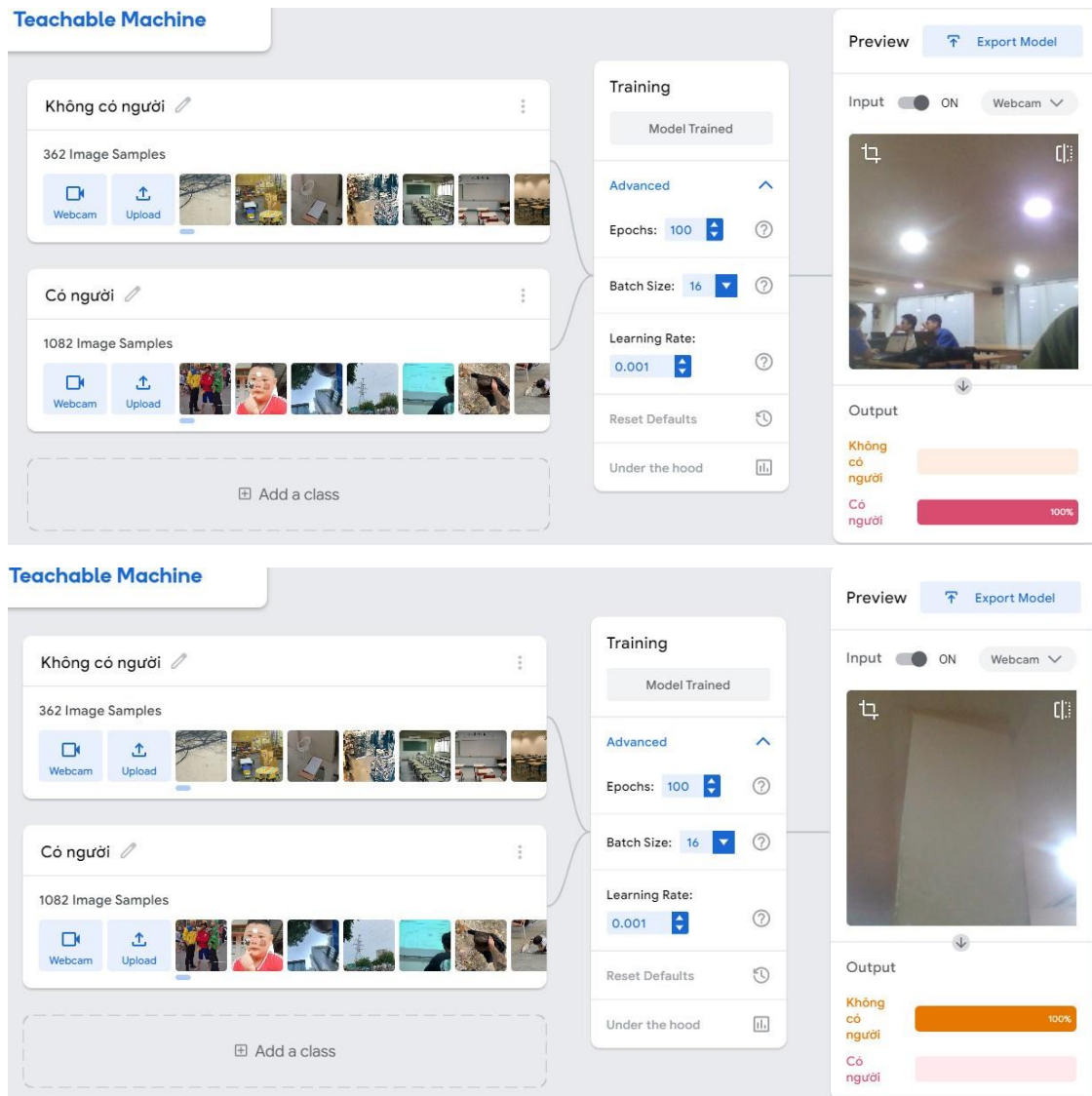
Input ON Webcam



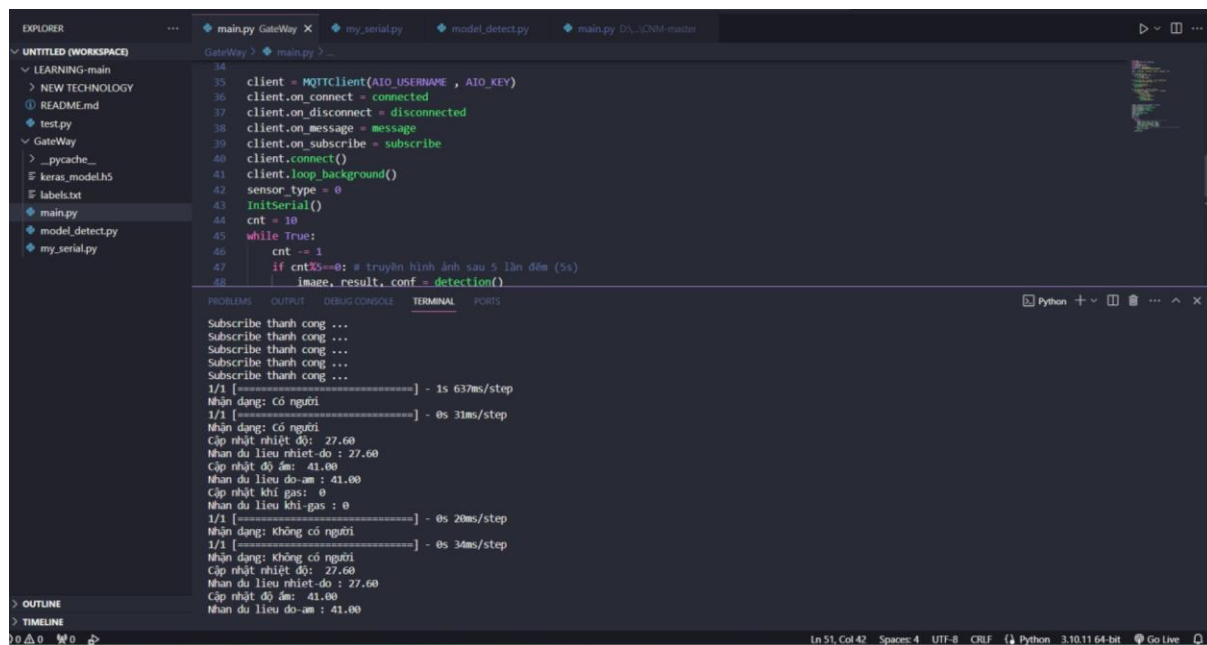
Output

Không có người

Có người 100%

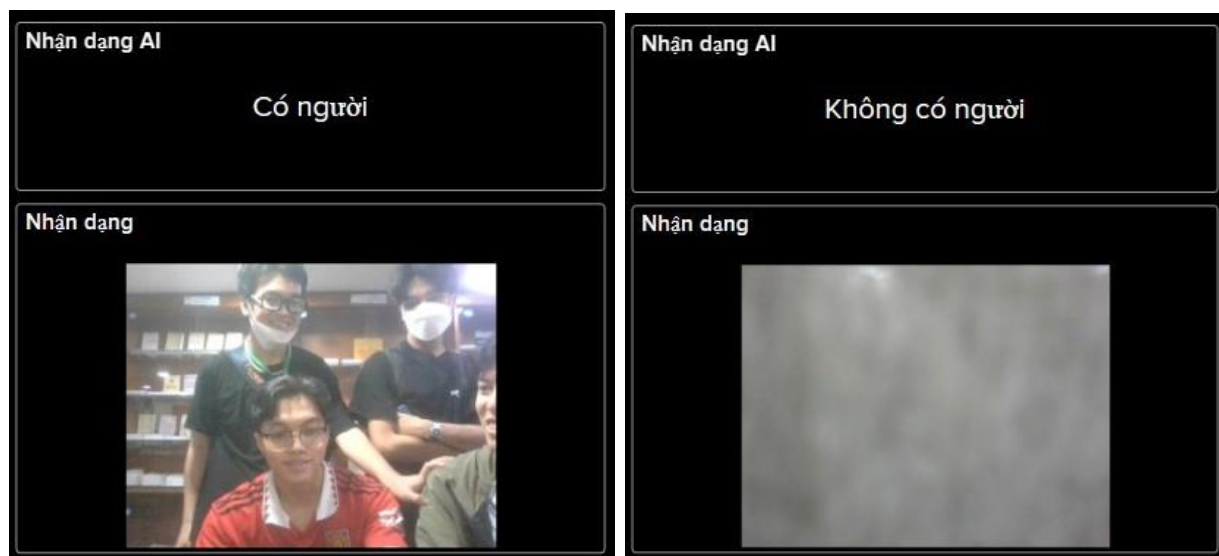


2.2. Hiện thực bằng Python



```
34
35 client = MQTTClient(AIO_USERNAME , AIO_KEY)
36 client.on_connect = connected
37 client.on_disconnect = disconnected
38 client.on_message = message
39 client.on_subscribe = subscribe
40 client.connect()
41 client.loop_background()
42 sensor_type = 0
43 InitSerial()
44 cnt = 10
45 while True:
46     cnt -= 1
47     if cnt==0: # truyền hình ảnh sau 5 lần đếm (5s)
48         image, result, conf = detection()
```

Subscribe thành công ...
Subscribe thành công ...
Subscribe thành công ...
Subscribe thành công ...
Subscribe thành công ...
1/1 [=====] - 1s 637ms/step
Nhận dạng: Có người
1/1 [=====] - 0s 31ms/step
Nhận dạng: Có người
Cập nhật nhiệt độ: 27.60
Nhận dữ liệu nhiệt độ: 27.60
Cập nhật độ ẩm: 41.00
Nhận dữ liệu độ ẩm: 41.00
Cập nhật khí gas: 0
Nhận dữ liệu khí gas: 0
1/1 [=====] - 0s 20ms/step
Nhận dạng: Không có người
1/1 [=====] - 0s 34ms/step
Nhận dạng: Không có người
Cập nhật nhiệt độ: 27.60
Nhận dữ liệu nhiệt độ: 27.60
Cập nhật độ ẩm: 41.00
Nhận dữ liệu độ ẩm: 41.00



3. Đầu nối

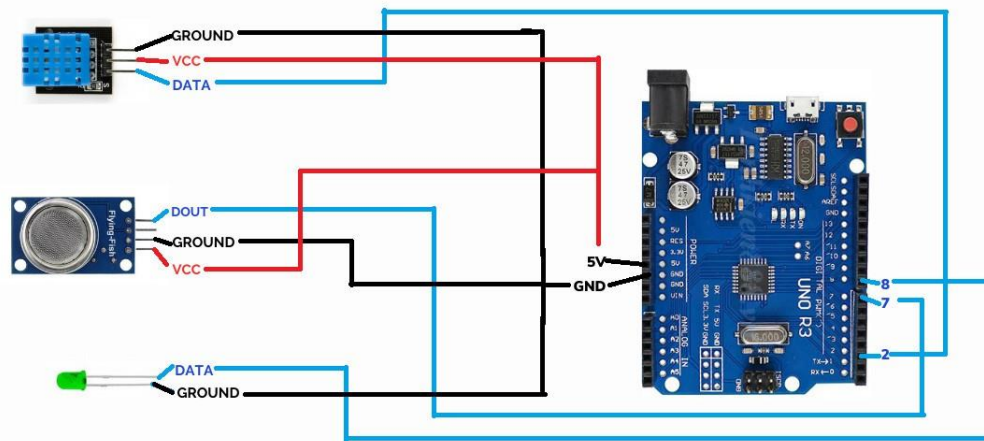
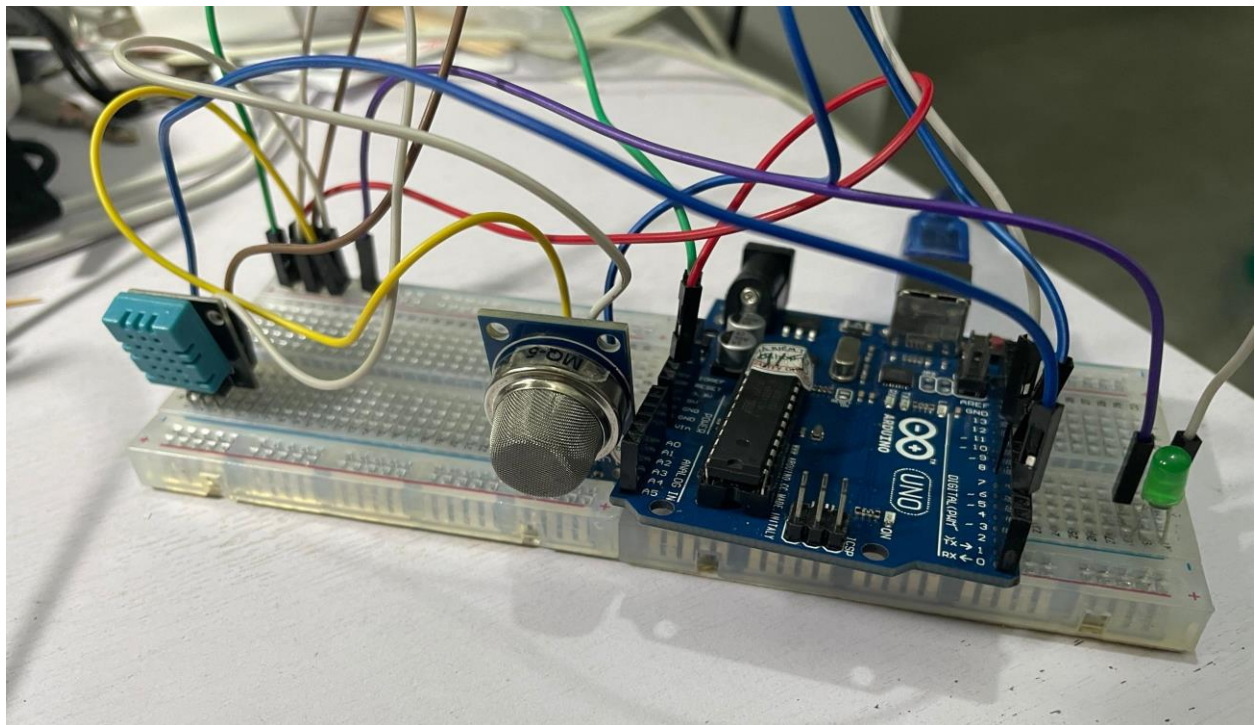
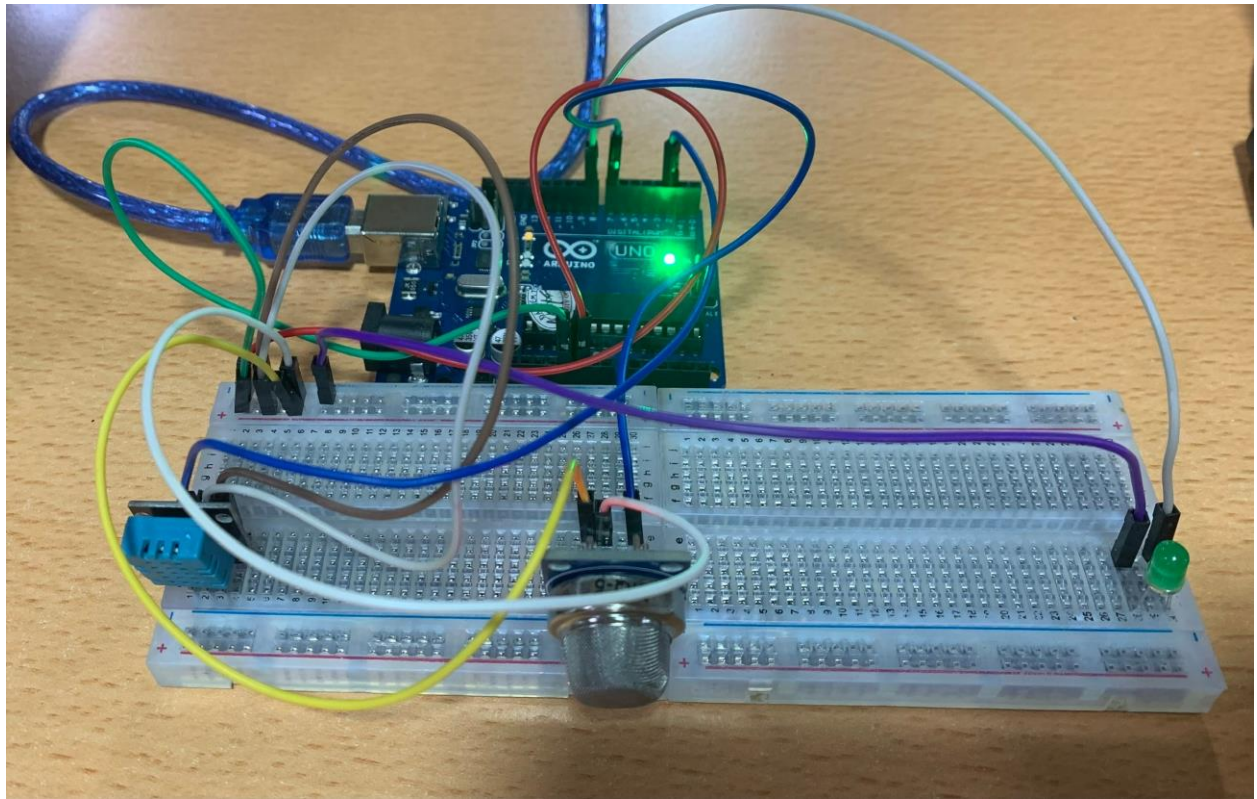


Figure 9. Sơ đồ đấu nối






```

2
3 #define DHTPIN 2
4 #define DHTTYPE DHT11
5
6 const int gasPin = 7;
7 const int ledPin = 8;
8
9 DHT dht(DHTPIN, DHTTYPE);
10
11 void setup() {
12     Serial.begin(9600);
13     dht.begin();
14     pinMode(ledPin, OUTPUT);
15 }
16
17 void loop() {
18     float humidity = dht.readHumidity();
19     float temperature = dht.readTemperature();
20     int ledState = digitalRead(ledPin);
21     int gasValue = digitalRead(gasPin);
22     // Đọc dữ liệu từ Serial và xử lý
23     if (Serial.available()) {
24         char data = Serial.read();
25         if (data == 'L') {
26             if (Serial.available()) {
27                 char stateChar = Serial.read();
28                 if (stateChar == '0') {
29                     digitalWrite(ledPin, LOW); // Tắt đèn LED
30                 } else if (stateChar == '1') {
31                     digitalWrite(ledPin, HIGH); // Bật đèn LED
32                 } else {
33                     digitalWrite(ledPin, LOW);
34                 }
35             }
36         } else {
37             return;
38         }
39     }
40     // Gửi dữ liệu nhiệt độ, độ ẩm và giá trị khí gas qua cổng Serial
41     Serial.print("T:");
42     Serial.print(temperature);
43     Serial.print(",H:");
44     Serial.print(humidity);
45     Serial.print(",G:");
46     Serial.println(gasValue);
47     delay(14000);
48 }
49

```

Figure 10. Lập trình nhúng thiết bị điện tử

4. Android Studio

4.1. Yêu cầu thiết bị

Phiên bản Android SDK:

- compileSdk = 34: Yêu cầu sử dụng phiên bản SDK 34 để biên dịch. Cần hỗ trợ phiên bản Android từ 11 (Android 11) trở lên.

Yêu cầu hệ điều hành:

- minSdk = 21: Yêu cầu tối thiểu Android 5.0 (API level 21).
- targetSdk = 21: Hoạt động tốt nhất trên Android 5.0 (API level 21).

Phiên bản Java: Java 8.

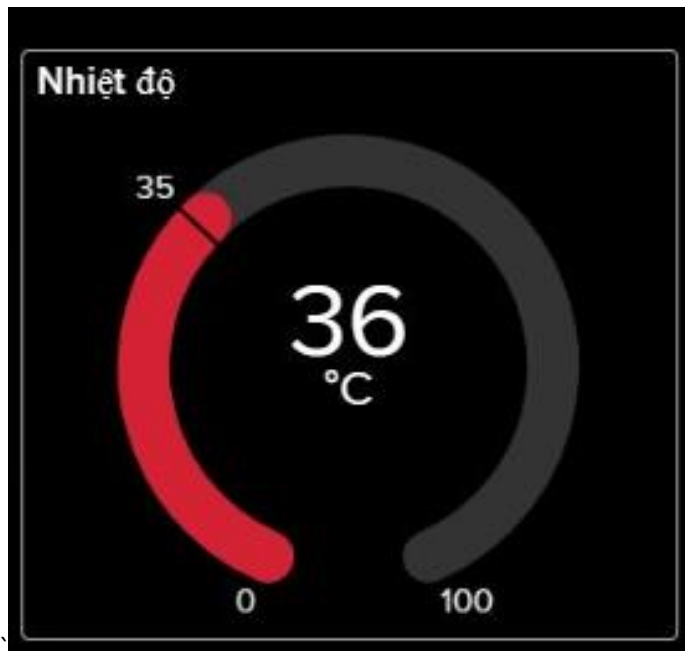
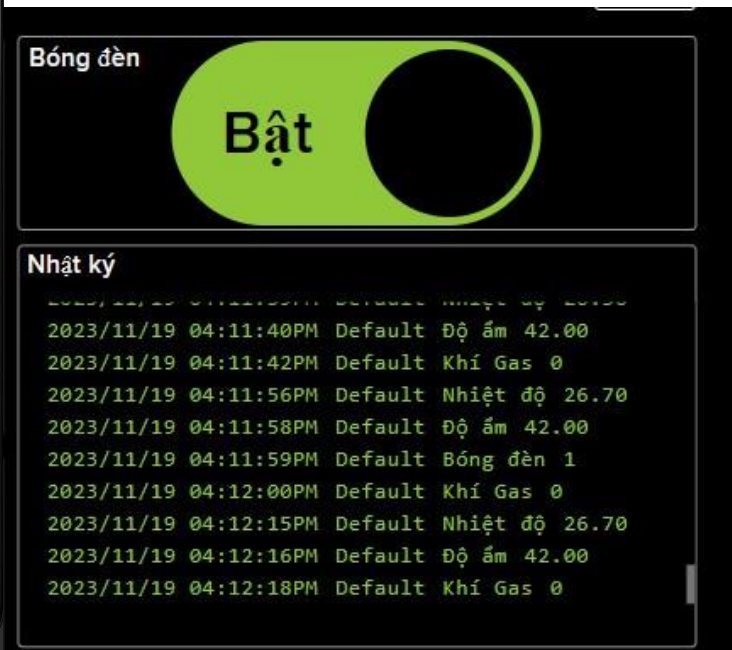
Dependencies:

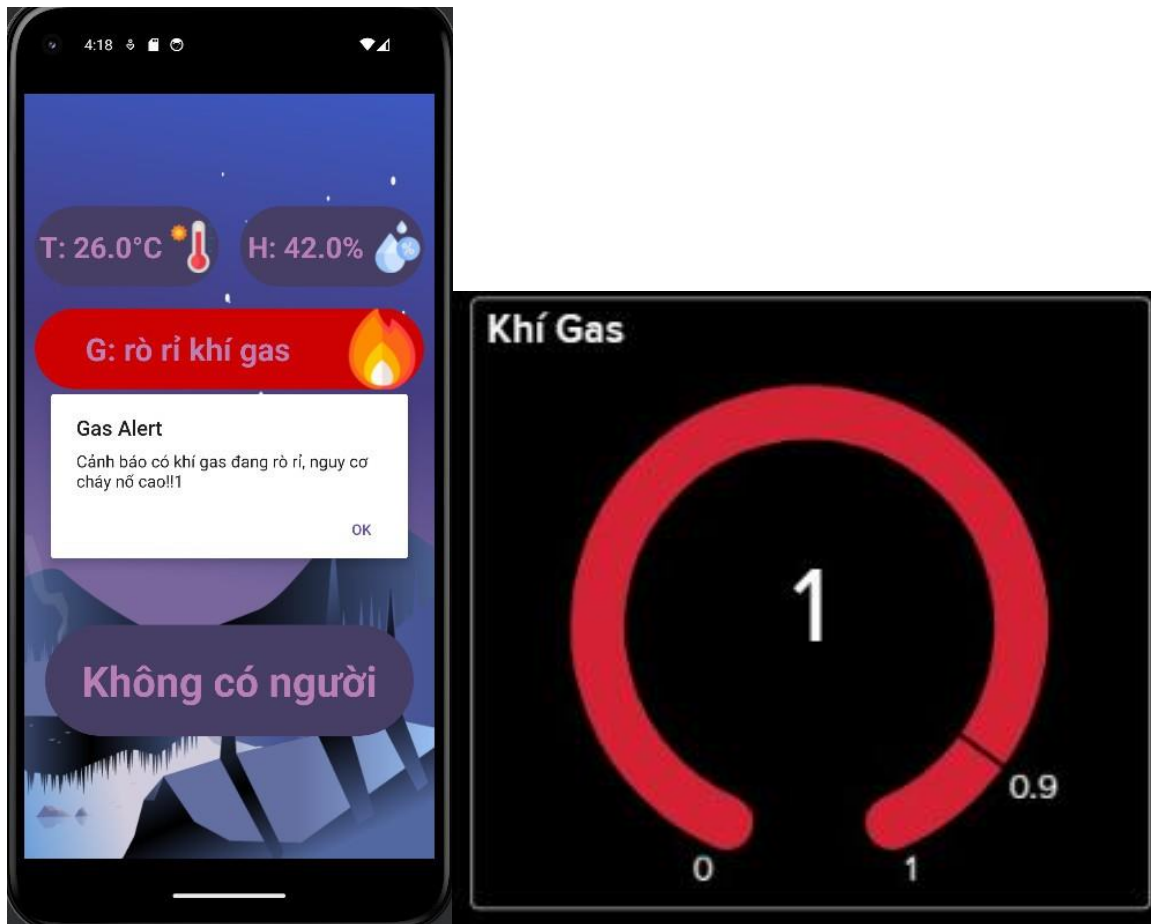
- Thư viện MQTT ver 1.1.0.

- Dịch vụ MQTT cho Android ver 1.1.1.
- Thư viện toggle ver 1.1.0.
- Thư viện hỗ trợ AppCompat ver 1.6.1.
- Thư viện Material Design ver 1.10.0.
- Thư viện ConstraintLayout ver 2.1.4.

4.2. Hiện thực bằng code







IV. Đánh giá và tổng kết

1. Mô hình nhận dạng

Dữ liệu đào tạo mô hình còn tương đối ít với trường hợp nhận dạng không có người.

Tuy nhiên, độ chính xác cao và độ ổn định tương đối tốt, nhưng còn phụ thuộc vào thiết bị và môi trường nhận dạng hình ảnh. (Môi trường thiếu ánh sáng, thiết bị nhận diện hình ảnh chất lượng kém thì sẽ cho ra kết quả có độ ổn định và chính xác không còn cao).

Tốc độ nhận dạng nhanh chóng và khả năng phân biệt tốt.

Vì sử dụng công cụ trực tuyến Teachable Machine nên mô hình hiện tại không có khả năng tùy chỉnh cao, tuy nhiên có thể tiến hành đào tạo lại một cách dễ dàng khi cần thiết.

2. Kết quả hiện thực

Đọc dữ liệu từ các thiết bị cảm biến lên Adafruit IO và truyền dữ liệu về thiết bị di động.

Chất lượng thiết bị điện tử còn hạn chế nên còn trục trặc khi hiện thực.

Tốc độ truyền dữ liệu giữa các thiết bị có độ trễ cao, dẫn đến việc nhận dữ liệu từ thiết bị cảm biến còn bị chậm trễ

V. Ứng dụng và hướng phát triển

Ứng dụng:

- Nhà thông minh, phát hiện và cảnh báo khả năng cháy nổ.

Hướng phát triển:

- Cải thiện tốc độ truyền và nhận dữ liệu trên các thiết bị cảm biến với thiết bị di động.
- Cải thiện chất lượng và tốc độ truyền hình ảnh lên các thiết bị. Đồng thời chuyển được hình ảnh thực lên được thiết bị di động.
- Phát triển mô hình ứng dụng hiện tại, thành mô hình thực tế lớn hơn, với khả năng thích ứng với nhiều thiết bị hơn là chỉ riêng Android Studio.

Tài liệu tham khảo:

Slides IOT:

https://lms.iuh.edu.vn/pluginfile.php/758101/mod_resource/content/1/Connect%20Adafruit%20IO.pdf

https://lms.iuh.edu.vn/pluginfile.php/760409/mod_resource/content/2/Stimulation.pdf

Adafruit IO:

<https://learn.adafruit.com/welcome-to-adafruit-io/io-faq>

<https://learn.adafruit.com/welcome-to-adafruit-io/adafruit-io-mqtt-api>

Teachable Machine:

<https://teachablemachine.withgoogle.com/faq>

AWS:

[What is MQTT? - MQTT Protocol Explained - AWS \(amazon.com\)](#)

Android Studio:

<https://developer.android.com/>

Linh kiện điện tử và đầu nối:

[Arduino UNO R3 DIP - Nshop \(nshopvn.com\)](#)

[Cảm biến khí hóa lỏng, khí than MQ-5 - Nshop \(nshopvn.com\)](#)

[Module Cảm Biến Độ Ẩm, Nhiệt Độ DHT11 - Nshop \(nshopvn.com\)](#)

[Led phủ màu 3mm - Nshop \(nshopvn.com\)](#)

Dữ liệu:

<https://www.kaggle.com/datasets/constantinwerner/human-detection-dataset>